Database Systems CSE 414

Lectures 4: Joins & Aggregation (Ch. 6.1-6.4)

Announcements

- WQ1 is posted to gradebook
 - double check scores
- WQ2 is out due next Sunday
- HW1 is due Tuesday (tomorrow), 11pm
- HW2 is coming out on Wednesday
- Should now have seats for all registered

Outline

- Inner joins (6.2, review)
- Outer joins (6.3.8)
- Aggregations (6.4.3 6.4.6)

UNIQUE

- PRIMARY KEY adds implicit "NOT NULL" constraint while UNIQUE does not
 - you would have to add this explicitly for UNIQUE:

```
CREATE TABLE Company(
  name VARCHAR(20) NOT NULL, ...
UNIQUE (name));
```

- You almost always want to do this (in real schemas)
 - SQL Server behaves badly with NULL & UNIQUE
 - otherwise, think through NULL for every query
 - you can remove the NOT NULL constraint later

```
SELECT a1, a2, ..., an FROM R1, R2, ..., Rm WHERE Cond
```

Company(<u>cname</u>, country)
Product(<u>pname</u>, price, category, manufacturer)
– manufacturer is foreign key

```
SELECT DISTINCT cname
```

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
Camera	Photo	Hitachi
OneClick	Photo	Hitachi

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer	
Gizmo	gadget	GizmoWorks	
Camera	Photo	Hitachi	
OneClick	Photo	Hitachi	

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

pname	category	manufacturer	cname	country
Gizmo	gadget	GizmoWorks	GizmoWorks	USA

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

_	pname	category	manufacturer	
	Gizmo	gadget	GizmoWorks	
	Camera	Photo	Hitachi	
	OneClick	Photo	Hitachi	

Company

	cname	country
_	GizmoWorks	USA
	Canon	Japan
	Hitachi	Japan

Not output since country != 'USA' (also cname != manufacturer)

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer	
Gizmo	gadget	GizmoWorks	
Camera	Photo	Hitachi	
OneClick	Photo	Hitachi	

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

Not output since country != 'USA'

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

	pname	category	manufacturer	
_	Gizmo	gadget	GizmoWorks	
l	Camera	Photo	Hitachi	
	OneClick	Photo	Hitachi	

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

Not output since category != 'gadget' (and ...)

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer	
Gizmo	gadget	GizmoWorks	
Camera	Photo	Hitachi	
OneClick	Photo	Hitachi	

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

Not output since category != 'gadget'

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

	pname	category	manufacturer	
_	Gizmo	gadget	GizmoWorks	
	Camera	Photo	Hitachi	
	OneClick	Photo	Hitachi	

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

Not output since category != 'gadget'

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
Camera	Photo	Hitachi
OneClick	Photo	Hitachi

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

Not output since category != 'gadget' (with any Company)

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
Camera	Photo	Hitachi
OneClick	Photo	Hitachi

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

restrict to category = 'gadget'

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product (where category = 'gadget')

pname	category	manufacturer
Gizmo	gadget	GizmoWorks

Company

cname	country
GizmoWorks	USA
Canon	Japan
Hitachi	Japan

restrict to country = 'USA'

SELECT DISTINCT cname

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Product (where category = 'gadget')

pname	category	manufacturer
Gizmo	gadget	GizmoWorks

Company (where country = 'USA')

cname	country
GizmoWorks	USA

Now only one combination to consider

(Query optimizers do this too.)

```
SELECT DISTINCT cname
```

FROM Product, Company

WHERE country = 'USA' AND category = 'gadget' AND

manufacturer = cname

Alternative syntax:

```
SELECT DISTINCT cname
```

FROM Product JOIN Company ON

country = 'USA' AND category = 'gadget' AND manufacturer = cname

Emphasizes that the predicate is part of the join.

Self-Joins and Tuple Variables

- Ex: find companies that manufacture both products in the 'gadgets' category and in the 'photo' category
- Just joining Company with Product is insufficient: need to join Company with Product with Product
 - FROM Company, Product, Product
- When a relation occurs twice in the FROM clause we call it a self-join; in that case every column name in Product is ambiguous (why?)
 - are you referring to the tuple in the 2nd or 3rd loop?

Name Conflicts

we used cname / pname to avoid this problem

- When a name is ambiguous, qualify it:
 - WHERE Company.name = Product.name AND ...
- For self-join, we need to distinguish tables:

```
FROM Product x, Product y, Company
```

- These new names are called "tuple variables"
 - can think of as name for the variable of each loop
 - can also write "Company AS C" etc.
 - can make SQL query shorter: C.name vs Company.name

SELECT DISTINCT z.cname

FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
SingleTouch	photo	Hitachi
MultiTouch	photo	GizmoWorks

cname	country
GizmoWorks	USA
Hitachi	Japan

SELECT DISTINCT z.cname

FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product

V		
X.		
•		_
		г

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
SingleTouch	photo	Hitachi

cname	country
GizmoWorks	USA
Hitachi	Japan

SELECT DISTINCT z.cname

FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product

)	<	

y

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
SingleTouch	photo	Hitachi
MultiTouch	photo	GizmoWorks

cname	country
GizmoWorks	USA
Hitachi	Japan

```
SELECT DISTINCT z.cname
```

FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product

MultiTouch

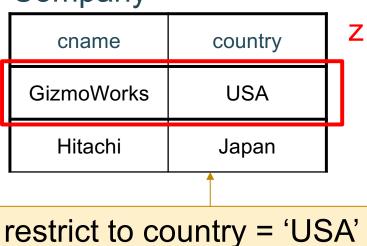
X

y

	pname	category	manutacturer
	Gizmo	gadget	GizmoWorks
L	SingleTouch	photo	Hitachi

photo

Company



Not output since y.category != 'photo'

GizmoWorks

```
FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product Company z
```

X	pname	category	manufacturer
	Gizmo	gadget	GizmoWorks
У	SingleTouch	photo	Hitachi
	MultiTouch	photo	GizmoWorks

cname	country	2	<u>Z</u>
GizmoWorks	USA		
Hitachi	Japan		

Company

Not output since y.manufacturer != cname

```
SELECT DISTINCT z.cname
```

FROM Product x, Product y, Company z

WHERE z.country = 'USA'

AND x.category = 'gadget'

AND y.category = 'photo'

AND x.manufacturer = cname

AND y.manufacturer = cname;

Product

X

pname	category	manufacturer
Gizmo	gadget	GizmoWorks
SingleTouch	photo	Hitachi
MultiTouch	photo	GizmoWorks

cname	country	7
GizmoWorks	USA	
Hitachi	Japan	

x.pname	x.category	x.manufacturer	y.pname	y.category	y.manufacturer	z.cname	z.country
Gizmo	gadget	GizmoWorks	MultiTouch	Photo	GizmoWorks	GizmoWorks	USA

Outer joins

Product(<u>name</u>, category)
Purchase(prodName, store) -- prodName is foreign key

```
SELECT Product.name, ..., Purchase.store
```

FROM Product, Purchase

WHERE Product.name = Purchase.prodName

Or equivalently:

```
SELECT Product.name, ..., Purchase.store
```

FROM Product JOIN Purchase ON

Product.name = Purchase.prodName

But some Products may not be not listed. Why?

Outer joins

Product(<u>name</u>, category)
Purchase(prodName, store) -- prodName is foreign key

If we want to include products that never sold, then we need an "outer join":

SELECT Product.name, ..., Purchase.store
FROM Product LEFT OUTER JOIN Purchase ON
Product.name = Purchase.prodName

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz

	Name	Category
\prod	Gizmo	gadget
	Camera	Photo
Ī	OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz

	Name	Category
_	Gizmo	gadget
	Camera	Photo
	OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz
Camera	Ritz

	Name	Category
_	Gizmo	gadget
	Camera	Photo
	OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

SELECT Product.name, Purchase.store FROM Product JOIN Purchase ON Product.name = Purchase.prodName

Product Purchase

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

SELECT Product.name, Purchase.store FROM Product LEFT OUTER JOIN Purchase ON Product.name = Purchase.prodName

Product Purchase

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

SELECT Product.name, Purchase.store FROM Product LEFT OUTER JOIN Purchase ON Product.name = Purchase.prodName

Product Purchase

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz

Name	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz
OneClick	NULL

SELECT Product.name, Purchase.store FROM Product RIGHT OUTER JOIN Purchase ON Product.name = Purchase.prodName

Product

Name	Category
Gizmo	gadget
Camera	Photo
OneClick	Photo

N	lame	Store
G	izmo	Wiz
Ca	amera	Ritz
Ca	amera	Wiz
N	IULL	Foo

Purchase

ProdName	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz
Phone	Foo

SELECT Product.name, Purchase.store FROM Product FULL OUTER JOIN Purchase ON Product.name = Purchase.prodName

Product Purchase

	_		_
Name		Category	
Gizmo		gadget	
Camera		Photo	
OneClick		Photo	
	Name		

Name	Store
Gizmo	Wiz
Camera	Ritz
Camera	Wiz
OneClick	NULL
NULL	Foo

ProdName Store Gizmo Wiz Camera Ritz Camera Wiz Phone Foo

Outer Joins

- Left outer join:
 - Include the left tuple even if there's no match
- Right outer join:
 - Include the right tuple even if there's no match
- Full outer join:
 - Include both left and right tuples even if there's no match
- (Also something called a UNION JOIN, though it's rarely used.)
- (Actually, all of these used much more rarely than inner joins.)

Outer Joins Example

See lec04-sql-outer-joins.sql...

Aggregation in SQL

```
>sqlite3 lecture04
sqlite> create table Purchase(
          pid int primary key,
          product text,
                                 Other DBMSs have
          price float,
                                   other ways of
          quantity int,
                                   importing data
          month varchar(15));
sqlite> -- download data.txt
sqlite> .import lec04-data.txt Purchase
```

Comment about SQLite

- One cannot load NULL values such that they are actually loaded as null values
- So we need to use two steps:
 - Load null values using some type of special value
 - Update the special values to actual null values

```
update Purchase
  set price = null
  where price = 'null'
```

Simple Aggregations

Five basic aggregate operations in SQL

```
select count(*) from Purchase
select sum(quantity) from Purchase
select avg(price) from Purchase
select max(quantity) from Purchase
select min(quantity) from Purchase
```

Except count, all aggregations apply to a single value

Aggregates and NULL Values

Null values are not used in aggregates

```
insert into Purchase
   values(12, 'gadget', NULL, NULL, 'april')
Let's try the following
    select count(*) from Purchase
    select count(quantity) from Purchase
    select sum(quantity) from Purchase
    select sum(quantity)
    from Purchase
    where quantity is not null;
```

Aggregates and NULL Values

Null values are not used in aggregates

```
insert into Purchase
   values(12, 'gadget', NULL, NULL, 'april')
Let's try the following
    select count(*) from Purchase
    select count(quantity) from Purchase
    select sum(quantity) from Purchase
    select sum(quantity)
    from Purchase
    where quantity is not null;
```

Counting Duplicates

COUNT applies to duplicates, unless otherwise stated:

```
SELECT Count(product)
FROM Purchase
WHERE price > 4.99
```

same as Count(*) if no nulls

We probably want:

```
SELECT Count(DISTINCT product)
FROM Purchase
WHERE price> 4.99
```

More Examples

SELECT Sum(price * quantity)
FROM Purchase

SELECT Sum(price * quantity)
FROM Purchase

WHERE product = 'bagel'

What do they mean?

Simple Aggregations

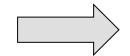
Purchase

Product	Price	Quantity
Bagel	3	20
Bagel	1.50	20
Banana	0.5	50
Banana	2	10
Banana	4	10

SELECT Sum(price * quantity)

FROM Purchase

WHERE product = 'Bagel'



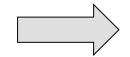
90 (= 60+30)

Simple Aggregations

Purchase

Product	Price	Quantity
Bagel	3	20
Bagel	1.50	20
Banana	0.5	50
Banana	2	10
Banana	4	10

SELECT Sum(price * quantity)
FROM Purchase
WHERE product = 'Bagel'



90 (= 60+30)

More Examples

How can we find the average revenue per sale?

```
SELECT sum(price * quantity) / count(*)
FROM Purchase
WHERE product = 'bagel'
```

How can we find the average price of a bagel sold?

```
SELECT sum(price * quantity) / sum(quantity)
FROM Purchase
WHERE product = 'bagel'
```

More Examples

```
SELECT sum(price * quantity) / count(*)
```

FROM Purchase

WHERE product = 'bagel'

```
SELECT sum(price * quantity) / sum(quantity)
```

FROM Purchase

WHERE product = 'bagel'

What happens if there are NULLs in price or quantity?

Moral: disallow NULLs unless you need to handle them